LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84
I l6r
no. 308-315
cop. 2

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

APR 26 1974

APR 9 Rec'd

L161—O-1096

Digitized by the Internet Archive
in 2013

http://archive.org/details/minimizationofsw309ibar

I l6n
no.309
cop.2

Report No. 309

MINIMIZATION OF SWITCHING NETWORKS USING NEGATIVE FUNCTIONS

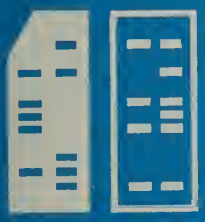
MAR 11 1969
MAR 17 1969

by

T. Ibaraki

S. Muroga

February 17, 1969



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

MINIMIZATION OF SWITCHING NETWORKS USING NEGATIVE FUNCTIONS

T. Ibaraki

S. Muroga

February 17, 1969

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Abstract

In this paper we discuss a minimization procedure of switching networks using negative functions only, assuming that any negative function can be implemented as a gate in the networks. The minimality criterion used is to minimize the number of gates in the network under the conditions:

(1) The network consists of two levels,

(2) No fan-ins restriction on each gate is imposed.

This is motivated by the recent progress in the MOS integrated circuit technology that a considerably complex negative function can be realized by a single MOS gate.

The minimization procedure is then extended to multiple output network design. Also some considerations are given to the problem of minimizing the number of connections and interconnections in the network.

Minimization of Switching Networks Using Negative Functions

1. Introduction

Recent advances in integrated circuit technology are changing the entire prospect of development of logical design. It opens a number of possibilities as well as complicates the logical design in many respects.

Especially, the MOS integrated circuit technology exhibits many attractive and practical features. Among those is an interesting fact, from which our present discussion has arisen, that each MOS gate is able to represent a considerably complex negative function because of its inherently high inpedance. For example, an MOS gate which represents any negative function with up to 32 literals in the functional form as

$$g = x_{11}\, x_{12} \cdots x_{1k_1} \vee x_{21}\, x_{22} \cdots x_{2k_2} \vee \cdots \vee x_{\ell 1}\, x_{\ell 2} \cdots x_{\ell k_\ell},$$

can be manufactured[9], as a typical implementation is shown in Fig. 1.[10]

In Fig.1, x_{ij} shows a connection from an external variable, and ϕ_1 and ϕ_2 are clock signal supplied from the external clock circuit.

As a consequence it is not too far from the reality to assume that any negative function can be realized with one gate, as far as a moderate number of input variables are considered.

With this observation, one might think of a new problem: minimization of the switching network composed of negative functions only, which realizes a given function (or functions). This problem is not trivial since each gate in the network is able to represent any negative function and we do not know in advance which of all possible negative functions each gate should represent.

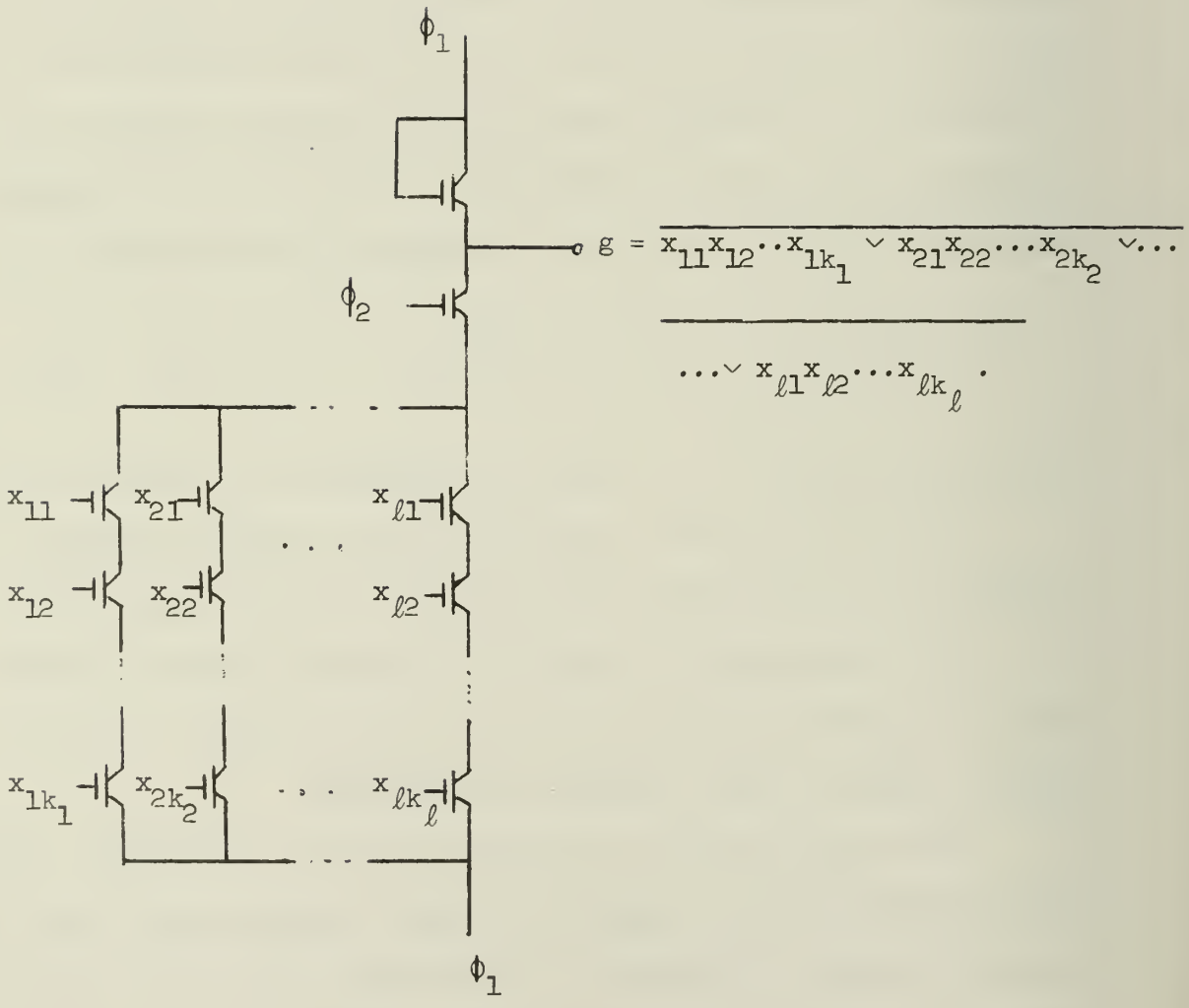


Fig. 1 A typical MOS gate .

As an example, consider a function of three variables x_1, x_2, x_3:

$$f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3. \tag{1.1}$$

One obvious design procedure of f with negative functions is first to realize three complemented variables $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$, using three gates respectively and then to realize f by regarding f as a negative function of x_1, x_2, x_3, $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$. This is illustrated in Fig. 2, in which each gate g_i represents a negative function.

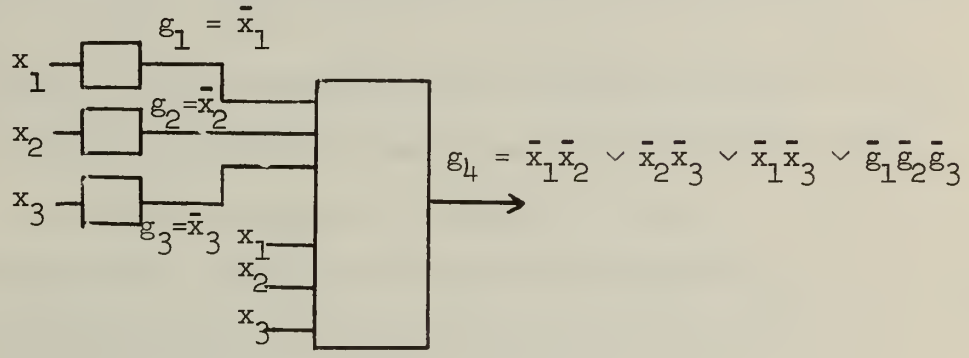


Fig. 2 Straightforward realization of $f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$

This synthesis procedure can be generalized to any function, indicating that any function of n variables can be realized with at most n+1 gates of negative functions within two levels.

The above network, however, can be simplified. A minimum network which is obtained by the method developed in this paper is shown in Fig. 3.

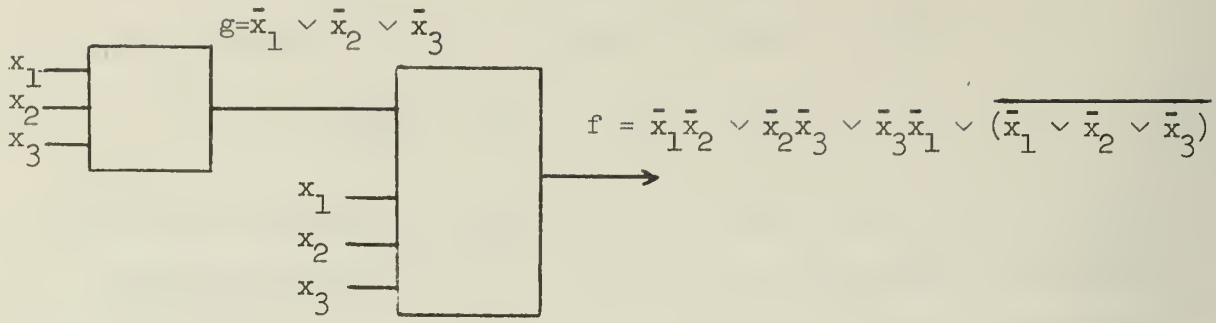


Fig. 3 Minimum realization of $f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$

Here, we mean by a minimum network a network with the minimum number of gates of negative functions under the conditions that;

(1) the given function f is realized within two levels (the level which consists of gates connected to the output gate is called the first level and the level of the output gate is the second level), and

(2) no fan-in restriction on each gate is imposed.

We will not minimize the number of interconnections in the network except some limited cases discussed in Section 7.

Although this minimization criterion may not exactly reflect the engineering requirements of the integrated circuit designers, this seems to be only criterion at present, which can be theoretically treated. It is to be explored whether or not a minimum network obtained under the above criterion would deviate much from those obtained under different minimality criteria. But we hope that the theory presented here would be a basis for further development.

2. Negative Functions

This section will define negative functions in terms of truth table. This will be a basis of the minimization procedure developed in this paper.

Let us first consider a completely specified function. A well-known definition of a negative function[5][8] is that a function is negative if and only if it has a disjunctive form with only complemented (i.e., negative) literals. For example, $\bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3$ is negative but $\bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$ is not. Since our procedure will be based on the monipulation of a truth table rather than Boolean expression, the following equivalent definition, which is also well-known[1][5][8] would be more useful.

Definition 1: A completely specified function h of m variables $\vec{y}_1, y_2, \ldots, y_m$ is negative if and only if there is no pair of input vectors $\vec{y}^{(j_1)}$ and $\vec{y}^{(j_2)}$ such that*

$$\vec{y}^{(j_1)} > \vec{y}^{(j_2)} \tag{2.1}$$

and

$$h(\vec{y}^{(j_1)}) = 1$$

$$h(\vec{y}^{(j_2)}) = 0, \tag{2.2}$$

where $\vec{y}^{(j_1)} > \vec{y}^{(j_2)}$ is defined as

$$y_i^{(j_1)} \geq y_i^{(j_2)} \quad \text{for every } i = 1, 2, \ldots, m,$$

and

$$y_i^{(j_1)} > y_i^{(j_2)} \quad \text{for at least one } i.$$

---

* Here we exclude from our consideration the contradictory situation such that

$$\vec{y}^{(j_1)} = \vec{y}^{(j_2)}$$

and

$$h(\vec{y}^{(j_1)}) = 1$$

$$h(\vec{y}^{(j_2)}) = 0.$$

This will never occur in our procedure.

As an example, the function f of variables x_1, x_2, x_3 shown in Table 1 (ignore the columns g_1, g_2, g_3 for a while) is not negative because there exists a pair such that

	f	x_1	x_2	x_3	g_1	g_2	g_3
(1)	1	0	0	0	1	1	1
(2)	1	0	0	1	1	1	
(3)	1	0	1	0	1		1
(4)	0	0	1	1	1		
(5)	1	1	0	0		1	1
(6)	0	1	0	1		1	
(7)	0	1	1	0			1
(8)	1	1	1	1	0	0	0

Table 1 Truth table of $f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$

$$\vec{x}^{(8)} = (111) > \vec{x}^{(4)} = (011)$$

and

$$f\,(111) = 1,\ f\,(011) = 0.$$

This function is the same function as that discussed in Fig. 1 and Fig. 2.

The outline of our procedure which realizes a given general (not necessarily negative) function by means of a combination of negative functions is now given. First we augment the truth table by several columns (illustrated as columns g_1, g_2, g_3 in Table 1) so that the given function f may be considered as a negative function if those new columns together with columns x_1, ..., x_n are regarded as variables of f. According to our procedure, each of these new columns (g_1, g_2, g_3 in Table 1) represents a negative function of variables x_1, x_2, ..., x_n. Therefore, from this augmented table, we can construct a network of negative functions

in which each gate in the first level corresponds to an augmented column (g_1, g_2, g_3 in Table 1) and the gate in the second level, to which the outputs of gates in the first level as well as the variables x_1, x_2, ..., x_n are fed, realizes the function f. (see Fig. 4.) In general, the augmented columns are incompletely specified, i.e., some entries of g_1, g_2, g_3 are left blank as seen in Table 1. Generation of augmented columns will be discussed later in detail.

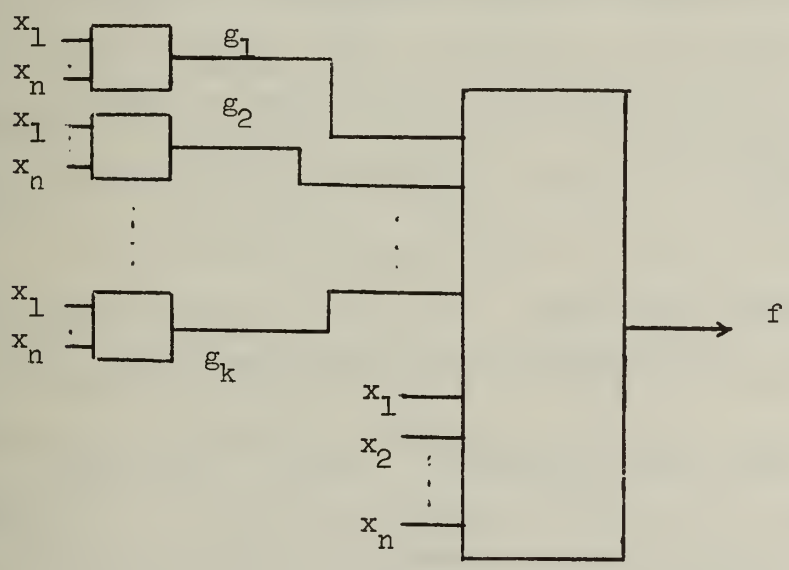


Fig. 4 General scheme of realization with negative functions

Throughout this paper let us assume that a function is generally incompletely specified.

As observed in Table 1, an incompletely specified function is possibly not specified in some of the output values and also in entries in some input vectors. For example, the output values of $g_1(x_1, x_2, x_3)$ in Table 1 are not specified for input vectors (100), (101) and (110). Also when we consider f as a function of input vectors $(x_1, x_2, x_3, g_1, g_2, g_3)$, then some of input vectors themselves are not completely specified. For example, the second row of g_3 in Table 1 is not specified. In this case, we interpret the output values of f to assume the values indicated in the column f, regardless of the assignments of unspecified entries. For example, the second row of Table 1 shows that the actual output value of f is 1, no matter whether the input vector is (001110) or (001111), i.e., regardless of the value in column g_3 which is blank in Table 1.

Now let us introduce a definition.

Definition 2: A column c_i is said negative with respect to $c_{i_1}, c_{i_2}, \ldots, c_{i_s}$ if and only if for every pair of entries $c_i^{(j_1)}$ and $c_i^{(j_2)}$ of c_i such that $c_i^{(j_1)} = 1$ and $c_i^{(j_2)} = 0$, there exists a subscript ℓ $(1 \leq \ell \leq s)$ which satisfies

$$c_{i_\ell}^{(j_1)} = 0$$

and

$$c_{i_\ell}^{(j_2)} = 1 . \qquad (2.3)$$

Consider the column f of Table 1. The column f is not negative with respect to x_1, x_2 and x_3 because for the pair of the eighth and the fourth entries, $f^{(8)} = 1$ and $f^{(4)} = 0$, there is no subscript ℓ $(1 \leq \ell \leq 3)$ such that $x_\ell^{(8)} = 0$ and $x_\ell^{(4)} = 1$. If x_1, x_2, x_3, g_1, g_2 and g_3 are considered as variables of f, however, then f is negative as easily verified. Note that the definition of the negativeness of a column is valid no matter whether related columns are completely specified or not.

In the following, first we will prove that two concepts of definitions 1 and 2, the negativeness of a function and that of a column, are equivalent if a completely specified case is concerned (Theorem 1). Then this equivalence will be extended to the incompletely specified case (Theorem 2). With those arguments, all the discussions throughout the following sections will be done in terms of the negativeness of columns.

The next definition of a negative function stated as Theorem 1 is a variant of Aker's theorem[2] which gave a necessary and sufficient condition for a function being a self-dual positive function. A comprehensive discussion is also found in [8].

Theorem 1 : A completely specified function h of m variables y_1, y_2, ..., y_m is negative if and only if the corresponding column h in the truth table is negative with respect to columns y_1, y_2, ..., y_m.

Proof: Let us prove that the condition for the column h to be negative with respect to columns $y_1, y_2, \ldots, y_m$ (Definition 2) is equivalent to the condition that there is no pair of input vectors of h, $\vec{y}^{(j_1)}$ and $\vec{y}^{(j_2)}$ such that $\vec{y}^{(j_1)} > \vec{y}^{(j_2)}$ and $h(\vec{y}^{(j_1)}) = 1$, $h(\vec{y}^{(j_2)}) = 0$ (Definition 1). Now note that $\vec{y}^{(j_1)} > \vec{y}^{(j_2)}$ is equivalent to non-existence* of any subscript ℓ such that $y_\ell^{(j_1)} = 0$ and $y_\ell^{(j_2)} = 1$. Therefore, if for every pair of entries, $h(\vec{y}^{(j_1)}) = 1$ and $h(\vec{y}^{(j_2)}) = 0$, there exists a subscript ℓ such that $y_\ell^{(j_1)} = 0$ and $y_\ell^{(j_2)} = 1$, it implies that there is no pair of vectors $\vec{y}^{(j_1)}$ and $\vec{y}^{(j_2)}$ such that $\vec{y}^{(j_1)} > \vec{y}^{(j_2)}$ and $h(\vec{y}^{(j_1)}) = 1$, $h(\vec{y}^{(j_2)}) = 0$, and vice versa. QED.

In order to extend Theorem 1 to incompletely specified functions, the next definition is introduced.

Definition 3: A completion of h is a completely specified function obtained from h by specifying all the unspecified output values of h. If a resultant completely specified function is negative, it is a negative completion.

Theorem 2: An incompletely specified function h of m variables $y_1, y_2, \ldots, y_m$ has a negative completion if and only if the column h in a truth table is negative with respect to columns $y_1, y_2, \ldots, y_m$.

Proof: The only-if part is trivial.

---

* $\vec{y}^{(j_1)} \nmid \vec{y}^{(j_2)}$ is assumed for every pair of j_1 and j_2, $j_1 \neq j_2$.

Now, consider an incompletely specified column h which is negative with respect to $y_1, y_2, \ldots, y_m$. First, if there are blank entries in columns $y_1, y_2, \ldots, y_m$, expand the table by assigning 0 and 1 to every blank entry, by muliplexing each row. (i.e. if a row is (01--1) where - denotes blank entries, replace this row by (01001), (01011), (01101), (01111).) As a result, the table has 2^m rows. The entries of the column h in the multiplexed rows of the new table are set to the same values as the original entries. This process takes care of input vectors which are incompletely specified. The column h in the new table is obviously still negative.

Next consider the case in which the entries in column h is incompletely specified. Denote the sets of true vectors and false vectors of h as follows:

$$A = \{ \vec{y}^{(j)} \mid h(\vec{y}^{(j)}) = 1\}$$
$$\text{and} \quad B = \{ \vec{y}^{(j)} \mid h(\vec{y}^{(j)}) = 0\}. \tag{2.4}$$

There is no pair of vectors $\vec{y}^{(j_1)}$ and $\vec{y}^{(j_2)}$ such that

$$\vec{y}^{(j_1)} > \vec{y}^{(j_2)} \text{ and } \vec{y}^{(j_1)} \in A, \ \vec{y}^{(j_2)} \in B,$$

from the negativeness of column h. Then pick up a vector $\vec{y}^{(k)}$ for which the value $h(\vec{y}^{(k)})$ is not specified yet. $\vec{y}^{(k)}$ satisfies one of the following three cases:

(1) There exists a vector $\vec{y}^{(p)}$ such that

$$\vec{y}^{(p)} \in A \text{ and } \vec{y}^{(p)} \geq \vec{y}^{(k)}. \tag{2.5}$$

(2) There exists a vector $\vec{y}^{(q)}$ such that

$$\vec{y}^{(q)} \in B \text{ and } \vec{y}^{(q)} \leq \vec{y}^{(k)}.$$

(3) None of (1) and (2) holds.

If case (1) occurs, then $h(\vec{y}^{(k)})$ is specified to 1. Then the resulting function h is characterized by its set of true vectors and set of false vectors;

$$A' = A \cup \{\vec{y}^{(k)}\}$$

and

$$B' = B.$$

The new column h' is still negative with respect to columns $y_1, y_2, \ldots, y_m$, because, otherwise the existence of vectors such that

$$\vec{y}^{(k)} > \vec{y}^{(j)}, \ \vec{y}^{(k)} \in A', \ \vec{y}^{(j)} \in B',$$

leads to a contradiction, i.e.,

$$\vec{y}^{(p)} > \vec{y}^{(j)}, \ \vec{y}^{(p)} \in A', \ \vec{y}^{(j)} \in B',$$

from condition (2.5).

If (2) is the case, then $h(\vec{y}^{(k)}) = 0$ is assigned. The proof is similarly obtained. Finally, if (3) is the case, either 0 or 1 is assigned to $h(\vec{y}^{(k)})$. Either case still keeps the column negative, even though one more entry is specified. This is because the non-negativeness of h' means the existence of

$$\vec{y}^{(k)} > \vec{y}^{(j)}, \ \vec{y}^{(k)} \in A', \ \vec{y}^{(j)} \in B'$$

(We assume $h'(\vec{y}^{(k)}) = 1$. The case of $h'(\vec{y}^{(k)}) = 0$ will be treated similarly.) which is a contradiction since this $\vec{y}^{(k)}$ satisfies the above case (2).

The assignment to the next unspecified entry of h' can be performed similarly. Repeating this process, we will eventually obtain a negative completion of h. QED.

As a result of Theorems 1 and 2, we learned that the negativeness of a column h with respect to columns $y_1, y_2, \ldots, y_m$ means that function h represented by column h in the truth table is a negative function (or has a negative completion) of variables $y_1, y_2, \ldots, y_m$. Thus the negativeness of completely specified function has been extended to the case of incompletely specified function.

Since the completion procedure described in the proof of Theorem 2 does not always yield the unique negative completion, any negative completion will be assumed in the following discussion because we are concerned with only negativeness of each function.

3. Realization of f With Negative Functions

If a given function f is not negative, more than one gate is needed to realize f with negative functions. As outlined in Section 2, our algorithm for the optimum network is based on the table manipulation. It first augments the given truth table with columns $g_1, g_2, \ldots, g_k$ such that

(1) the column f is negative with respect to $x_1, x_2, \ldots, x_n, g_1, \ldots, g_k$.

(2) the columns g_i, i = 1, 2, ..., k, are negative with respect to $x_1, x_2, \ldots, x_n$.

Provided that those columns $g_1, g_2, \ldots, g_k$ with the above properties are obtained, the corresponding network for f with negative functions is shown in Fig. 4. Each gate is the first level corresponds to one of g_i's, and the gate in the second level realizes f as a function of $x_1, x_2, \ldots, x_n, g_1, \ldots, g_k$.

In this section, we will discuss how to obtain $g_1, g_2, \ldots, g_k$ which satisfy the above requirements. The minimization of the number of gates will be discussed in the subsequent sections.

Definition 4: A pair of entries $c_i^{(j_1)} = 1$ and $c_i^{(j_2)} = 0$ in a column c_i is unallowable with respect to $c_{i_1}, c_{i_2}, \ldots, c_{i_s}$ if and only if there is no subscript ℓ ($1 \leq \ell \leq s$) such that

$$c_{i_\ell}^{(j_1)} = 0 \text{ and } c_{i_\ell}^{(j_2)} = 1. \tag{3.1}$$

If there exists a subscript ℓ which satisfies (3.1), $c_i^{(j_1)}$ and $c_i^{(j_2)}$ is said allowable due to c_{i_ℓ}.

Note that the column c_i is negative with respect to columns $c_{i_1}, c_{i_2}, \ldots, c_{i_s}$ if and only if any pair of entries of c_i such that $c_i^{(j_1)} = 1$ and $c_i^{(j_2)} = 0$ is allowable.

Now let us form the negativity table of function f of n variables $x_1, x_2, \ldots, x_n$. The negativity table is constructed according to the following procedure:

(1) Exhaust all unallowable pairs of column f with respect to columns $x_1, x_2, \ldots, x_n$, in the truth table of f.

(2) For each unallowable pair of entries, $f(\vec{x}^{(j_1)}) = 1$ and $f(\vec{x}^{(j_2)}) = 0$, add a new column e_j whose entries are $e_j^{(j_1)} = 0$, $e_j^{(j_2)} = 1$ and blank in all other entries. (i.e. e_j has exactly two entries which are not blank.) e_j is called a primitive additional column of f.

For example, the function f shown in Table 1 has three unallowable pair of entries of f, as easily seen. For these three unallowable pairs, three columns e_1, e_2, e_3 are added to the truth table. The negativity table of f which results is shown in Table 2.

f	x_1	x_2	x_3	e_1	e_2	e_3
1	0	0	0			
1	0	0	1			
1	0	1	0			
0	0	1	1	1		
1	1	0	0			
0	1	0	1		1	
0	1	1	0			1
1	1	1	1	0	0	0

Table 2 Negativity table of $f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$

Because of the way each e_j being generated, the next theorem is obvious.

Theorem 3: Let e_1, e_2, ..., e_t be all the primitive additional columns generated for the negativity table of f of n variables x_1, x_2, ..., x_n. Then f is negative with respect to x_1, x_2, ..., x_n, e_1, e_2, ..., e_t.

Another property of negativity table is given next.

Theorem 4: Each primitive additional column e_j of a negativity table is negative with respect to columns x_1, x_2, ..., x_n.

Proof: Assume that e_j is added corresponding to the unallowable pair, $f(\vec{x}^{(j_1)}) = 1$ and $f(\vec{x}^{(j_2)}) = 0$, to form the negativity table of f.

Then $e_j^{(j_1)} = 0$ and $e_j^{(j_2)} = 1$, There is no ℓ such that $x_\ell^{(j_1)} = 0$ and $x_\ell^{(j_2)} = 1$. Hence there must be a subscript r ($1 \leq r \leq n$) such that

$$x_r^{(j_1)} = 1 \text{ and } x_r^{(j_2)} = 0. \tag{3.2}$$

Condition (3.2) shows that column e_j is negative with respect to columns $x_1, x_2, \cdots, x_n$. QED.

Theorems 3 and 4 indicate that each e_j can be used as a negative function g_j in Fig. 4. Of course, this realization needs (t + 1) negative functions, where t is the number of columns e_j, and may not be minimum.

Definition 5: A column c_i is said to cover column c_j if and only if for every entry $c_j^{(k)}$ which is specified,

$$c_i^{(k)} = c_j^{(k)}$$

holds. (if an entry of $c_j^{(k)}$ is not specified, the above equality does not necessarily hold.)

For example, in Table 3, c_1 covers c_2 but does not cover c_3. c_3 covers neither c_1 nor c_2, etc.

c_1	c_2	c_3
1		1
0	0	
		0
1	1	1

Table 3 Illustration of the covering relation

The motivation of introducing this concept lies in the fact that if a column g covers a column e_j of the negativity table and is negative with respect to columns $x_1, x_2, \ldots, x_n$, then we can replace e_j by g. (i.e. f is negative with respect to $x_1, x_2, \ldots, x_n, e_1, \ldots, e_{j-1}, g, e_{j+1}, \ldots, e_t$ and g is negative with respect to $x_1, x_2, \ldots, x_n$.)

Hence, if g covers more than one e_j, all these e_j can be replaced by g reducing the number of negative functions in the network as shown in Fig. 4.

Again, the function f given in Tables 1 and 2 is considered. For this function f, assume the column as shown in Table 4.

f	x_1	x_2	x_3	M
1	0	0	0	
1	0	0	1	
1	0	1	0	
0	0	1	1	1
1	1	0	0	
0	1	0	1	1
0	1	1	0	1
1	1	1	1	0

Table 4

Realization of $f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee x_1x_2x_3$

As easily checked, M is negative with respect to x_1, x_2, x_3 and moreover M covers e_1, e_2, e_3 shown in Table 2. Therefore, all the three negative functions e_1, e_2, e_3 of variables x_1, x_2, x_3 can be replaced by a single negative function M of variables x_1, x_2, x_3. Fig. 2 will result by using the negative completion of M (see Table 5 in the next section);

$$M^* = \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3.$$

In the next section, a systematic generation of the minimum number of negative functions which together cover all of e_1, e_2, ..., e_t in the negativity table will be discussed.

4. Minimum Network by Maximal Compatible Sets

Let us begin with a few new definitions.

Definition 6: The conjoint of two columns c_i and c_j, denoted as $c_\ell = c_i \nabla c_j$, is defined by the following rule for each r:

$$c_\ell^{(r)} = c_i^{(r)} \nabla c_j^{(r)},$$

where $c_i^{(r)} \nabla c_j^{(r)}$ is given by

$c_j^{(r)}$ \ $c_i^{(r)}$	1	0	
1	1	?	1
0	?	0	0
	1	0	

The mark ? in the table shows that the operation is not defined. And blank entry denotes unspecified entry.

If all entries of c_ℓ are defined, (i.e. 1, 0 or blank but not ?) c_ℓ is said definable. The conjoint of more than two columns can be defined by the following inductive formula:

$$c_{i_1} \nabla c_{i_2} \nabla \dots \nabla c_{i_k} = (c_{i_1} \nabla c_{i_2} \nabla \dots \nabla c_{i_{k-1}}) \nabla c_{i_k}.$$

Definition 7: Two columns c_i and c_j are compatible with respect to c_{ℓ_1}, c_{ℓ_2}, ..., c_{ℓ_s} if and only if $c_i \nabla c_j$ is definable and also negative with respect to c_{ℓ_1}, c_{ℓ_2}, ..., c_{ℓ_s}.

We will examine the compatibility between each pair of columns, e_i and e_j, in the negativity table.

Definition 8: Consider a set of columns

$$C_i = \{ c_{i_1}, c_{i_2}, \ldots, c_{i_k} \} . \tag{4.1}$$

If and only if every two columns in C_i are compatible, with respect to a set of columns B, C_i is called compatible with respect to B. A compatible set C_i covers a column c_j if and only if $c_{i_1} \nabla c_{i_2} \nabla \ldots \nabla c_{i_k}$ covers c_j.

C_i of (4.1) is essentially $c_{i_1} \nabla c_{i_2} \nabla \ldots \nabla c_{i_k}$ according to the following lemma.

Lemma 1 C_i defined by (4.1) is compatible with respect to a set of columns B if and only if $c_{i_1} \nabla c_{i_2} \nabla \ldots \nabla c_{i_k}$ is definable and negative with respect to B.

The proof is trivial and hence omitted.

Definition 9: Consider all the compatible sets with respect to B. A compatible set C_i is said maximal if, and only if among these sets, there is no other compatible set with respect to B, which includes C_i.

Throughout this paper we will deal with only compatible sets or maximal compatible sets of primitive additional columns $e_1, e_2, \ldots, e_t$ in the negativity table with respect to $x_1, x_2, \ldots, x_n$. Therefore, we will refer to these simply as compatible sets or maximal compatible sets by dropping the phrase "of primitive additional columns $e_1, e_2, \ldots, e_t$ in the negativity table with respect to $x_1, x_2, \ldots, x_n$."

For example, e_1, e_2 and e_3 of Table 2 are pairwise compatible. Hence $M = \{ e_1, e_2, e_3 \}$ is a compatible set. Obviously this is maximal.

The concept of maximal compatible sets is important because of the next theorem.

Theorem 5: There exists a network with the minimum number of gates for a given function f in which every gate represents a negative function and those in the first level represent negative completions of maximal compatible sets.

Proof: Consider the negativity table of f. $e_1, e_2, \ldots, e_t$ are all primitive additional columns. Assume that the realization as shown in Fig. 4 is minimum, i.e., k gates in the first level represent $g_1, g_2, \ldots, g_k$ and (k+1) is the minimum number of gates which can realize f. Because the last gate represents a negative function with respect to $g_1, g_2, \ldots, g_k$, $x_1, x_2, \ldots, x_n$, any e_j in the negativity table must be covered by some of g_i. Now let us suppose that

$$g_1 \text{ covers } e_{11}, e_{12}, \ldots, e_{1j_1}$$

$$g_2 \text{ covers } e_{21}, e_{22}, \ldots, e_{2j_2}$$

$$\vdots$$

$$g_k \text{ covers } e_{k1}, e_{k2}, \ldots, e_{kj_k}.$$

Then since each g_i is negative with respect to $x_1, x_2, \ldots, x_n$, and covers $e_{i1} \nabla e_{i2} \nabla \ldots \nabla e_{ij_i}$, $\{e_{i1}, e_{i2}, \ldots, e_{ij_i}\}$ is a compatible set for $i = 1, 2, \ldots, k$ (Lemma 1). Therefore for every $\{ e_{i1}, e_{i2}, \ldots, e_{ij_i} \}$, $i = 1, 2, \ldots, k$, it is possible to consider a maximal compatible set which includes $\{e_{i1}, e_{i2}, \ldots, e_{ij_i}\}$.

respectively. Let us call them as $M_1, M_2, \ldots, M_k$. Finally we can replace each g_i in the network by a negative completion of M_i. After this replacement, f is still negative with respect to $M_1^*, M_2^*, \ldots, M_k^*, x_1, x_2, \ldots, x_n$, where M_i^* is a negative completion of M_i. This network also gives a minimum network since the new network uses the same number of gates as the network of $g_1, g_2, \ldots, g_k$ which is minimum. QED.

Because of this theorem, the synthesis of a negative function network is done by first generating all the maximal compatible sets and then selecting the minimum number of sets from them, which together cover all the primitive additional columns $e_1, e_2, \ldots, e_t$. The generation and selection of maximal compatible sets will be discussed in the next section.

Let us consider the previous example given in Table 1. Table 2 is the negativity table of f. We have seen that $M = \{ e_1, e_2, e_3\}$ is the maximal compatible set and no other maximal compatible set exists. M is the one shown in Table 4. Now, we can realize f with two gates, one is for M and the other is the last gate which gives f. A negative completion of M, M^*, is shown in Table 5.

f	x_1	x_2	x_3	M^*
1	0	0	0	1^*
1	0	0	1	1^*
1	0	1	0	1^*
0	0	1	1	1
1	1	0	0	1^*
0	1	0	1	1
0	1	1	0	1
1	1	1	1	0

Table 5

Completion of M

The entries of M^* with stars were newly specified, following the procedure given in the proof of Theorem 2. (All entries were determined uniquely by cases of (1) and (2) of the proof of Theorem 2.) M^* is written as

$$M^* = \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 .$$

Note that f is now incompletely specified if it is regarded as a function of x_1, x_2, x_3 and M^*. A negative completion easily obtained by inspection is

$$f = \bar{x}_1\bar{x}_2 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_3 \vee \bar{M}^* .$$

These completions of M and f give the network shown in Fig. 2. The procedure described so far explains how the minimum network of Fig. 2 was synthesized.

5. Generation and Selection of Maximal Compatible Sets

Let us start this section with a new example whose negativity table is shown in Table 6.

f	x_1	x_2	x_3	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
0	0	0	0	1	1	1	1				
1	0	0	1	0							
0	0	1	0					1	1		
0	0	1	1							1	
1	1	0	0		0						
0	1	0	1								1
1	1	1	0			0		0			
1	1	1	1				0		0	0	0

Table 6

Negativity table of $f = x_1x_2 \vee x_1\bar{x}_3 \vee \bar{x}_1\bar{x}_2x_3$

From primitive additional columns $e_1, e_2, \ldots, e_t$ specified in a negativity table, we will construct the compatibility matrix of f. The compatibility matrix is a triangular matrix in which columns correspond to $e_1, e_2, \ldots, e_{t-1}$ and rows correspond to $e_2, e_3, \ldots, e_t$, as shown in Table 7. A cross is entered if e_i and e_j which correspond to the row and the column of the entry are not compatible. Otherwise it is left as blank. The compatibility matrix of f derived from Table 6 is given in Table 7.

e_2							
e_3							
e_4							
e_5							
e_6							
e_7	X						
e_8	X	X					
	e_1	e_2	e_3	e_4	e_5	e_6	e_7

Table 7

Compatibility matrix of $f = x_1x_2 \vee x_1\bar{x}_3 \vee \bar{x}_1\bar{x}_2x_3$

The compatibility matrix tells us that all pairs of primitive additional columns e_i and e_j are compatible except three pairs (e_1,e_7), (e_1,e_8) and (e_2,e_8).

With the aid of compatibility matrix, we can generate all the maximal compatible sets in the following way. Let us consider primitive additional columns e_1, e_2, ..., e_t.

(1) Obtain all the maximal compatible sets of e_1 only. Obiviously there is only one maximal compatible set $\{e_1\}$.

(2) Given all the maximal compatible sets M_1, M_2, ..., M_b of e_1, e_2, ..., e_k $(1 \leq k \leq t-1)$, introduce e_{k+1} and form the sets according to the following rule for each i, i = 1, 2, ..., b.

(a) If the set $M_i \cup \{e_{k+1}\}$ is compatible, i.e., e_{k+1} is compatible with every element of M_i, then form

$$M_i' = M_i \cup \{e_{k+1}\} ,$$

(b) Otherwise let $\hat{M}_i$ be the set of all the elements of M_i which are compatible with e_{k+1}. Then form two new sets:

$$M_i'' = M_i ,$$

$$M_i''' = \hat{M}_i \cup \{e_{k+1}\} .$$

(3) Delete all the sets which are included in some other sets generated by (2) above. Note that only M_i''' could be included in other set. The remaining sets are all the maximal compatible sets of $e_1, e_2, \ldots, e_{k+1}$.

(4) Increase k by one. If the new k is equal to t, the procedure is completed, otherwise go to (2) again.

In the above recursive procedure, it is not obvious whether all the maximal compatible sets of $e_1, e_2, \ldots, e_{k+1}$ are generated after completing (3). This will be affirmatively proved in Theorem 6. For illustration let us obtain all the maximal compatible sets of $e_1, e_2, \ldots, e_8$ whose compatibility relations are given in Table 7. The generation procedure is illustrated in Fig. 5.

Fig. 5 shows that only one maximal compatible set is generated for each k not more than 6, since any two of $e_1, e_2, \ldots, e_6$ are compatible as seen from Table 7. When e_7 is added, two maximal compatible sets result

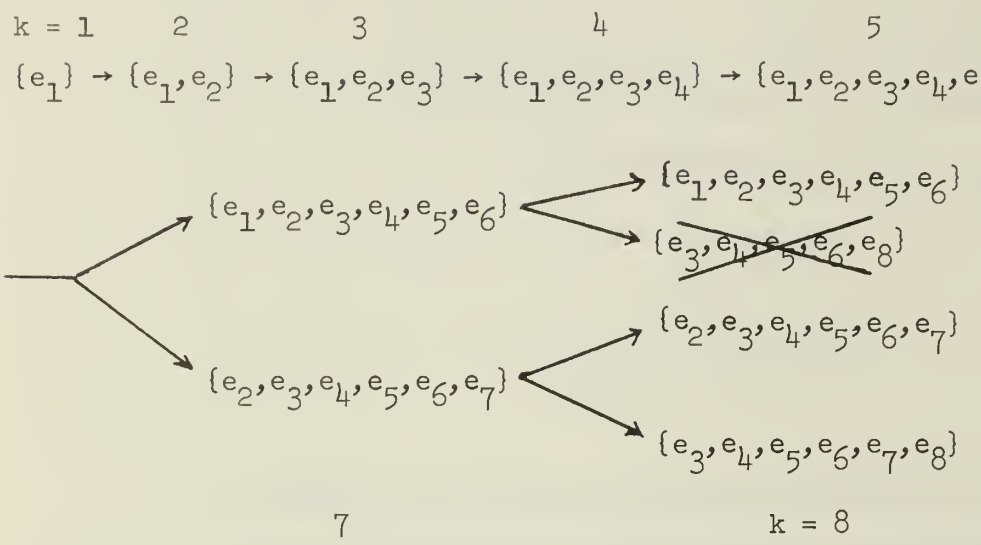


Fig. 5 Generation of maximal compatible sets of f given by Table 6.

since procedure 2(b) is applied because of non-compatibility of e_1 with e_7. Finally for k = 8, three maximal compatible sets are obtained. $\{e_3, e_4, e_5, e_6, e_8\}$ is deleted because this is included in $\{e_3, e_4, e_5, e_6, e_7, e_8\}$.

Theorem 6: Assume that M_1, M_2, ..., M_b are all the maximal compatible sets of e_1, e_2, ..., e_k. All the sets obtained as a consequence of the above steps (2) and (3) are maximal compatible sets of e_1, e_2, ..., e_{k+1}. Moreover there is no other maximal compatible set of $e_1, e_2, \ldots, e_{k+1}$.

Proof: To prove that the above procedure generate all the maximal sets, let us assume that M is a maximal compatible set of $e_1, e_2, \ldots, e_{k+1}$ and is not generated by steps (2) and (3) in the above procedure. M must contain e_{k+1} because otherwise M is equal to one of M_1, M_2, ..., M_b. Let M' be M from which e_{k+1} is deleted. Then M' is included in some of $M_1, M_2, \ldots, M_b$ because any two elements in M' are compatible. Let $M_i \supseteq M'$ for some i of i = 1, 2, ...b. Then M' must be included in or equal to M_i' or M_i''' of step (2). This is a contradiction with the initial assumption that M is maximal and is not generated by steps (2) and (3). Next it is obvious that after step (3) no compatible set which is not maximal remains. QED.

Definition 10: A minimal compatible cover of $e_1, e_2, \ldots, e_t$ for f is a set of the minimum number of maximal compatible sets such that every e_i belongs to at least one of maximal compatible sets chosen.

Now note that to find a minimal compatible cover is equivalent to designing a minimal network of negative functions. However, finding a minimal compatible cover is nothing but a set-covering problem which has been discussed in the literature in conjunction with a variety of problems.

Let $M_1, M_2, \ldots, M_p$ be all the maximal compatible sets. The covering table is a table, the i th row of which corresponds to e_i and the j th column of which corresponds to M_j. The (i,j) th entry of the covering table is 1 if e_i is covered by M_j, and blank otherwise. The set-covering problem is to find the minimum number of columns so that every row has an entry 1 in at least one of the selected columns. A number of techniques to solve the set-covering problem through the manipulation of the covering table are discussed in [7], for example.

Another interesting approach is the integer programming approach. The formulation of a set covering problem by an integer programming is given in [3][4], for example. Recent computational result[6] shows that some problems which correspond* to the cases (30 rows, 50 columns) and (30 rows, 90 columns) are solved in 0.15 seconds and 1.60 seconds respectively on the IBM 360/75I computer.

The covering table for the above example (Table 6 and Fig. 5) is now shown as an example.

Let

$$M_1 = \{e_1, e_2, e_3, e_4, e_5, e_6\}$$

$$M_2 = \{e_2, e_3, e_4, e_5, e_6, e_7\}$$

$$M_3 = \{e_3, e_4, e_5, e_6, e_7, e_8\} .$$

Then the covering table is:

	M_1	M_2	M_3
e_1	1		
e_2	1	1	
e_3	1	1	1
e_4	1	1	1
e_5	1	1	1
e_6	1	1	1
e_7		1	1
e_8			1

Table 8

Covering table of the example given by Table 6 and Fig. 5.

* The correspondence is not exact in the sense that the matrices are generated randonly and may not represent switching functions.

Obviously M_1 and M_3 constitute the minimal compatible cover of $e_1, e_2, \ldots, e_8$ which is unique in this particular example.

When M_1 and M_3 are considered as functions of variables x_1, x_2, x_3, their negative completions M_1^*, M_3^* are given in Table 9.

f	x_1	x_2	x_3	M_1^*	M_3^*
0	0	0	0	1	1
1	0	0	1	0	1*
0	0	1	0	1	1
0	0	1	1	0*	1
1	1	0	0	0	1*
0	1	0	1	0*	1
1	1	1	0	0	0
1	1	1	1	0	0

Table 9

Completion of M_1 and M_3

M_1^* and M_3^* are written as

$$M_1^* = \bar{x}_1 \bar{x}_3$$

$$M_3^* = \bar{x}_1 \vee \bar{x}_2 .$$

The minimum network of f using M_1^* and M_3^* as gates in the first level is given in Fig. 6.

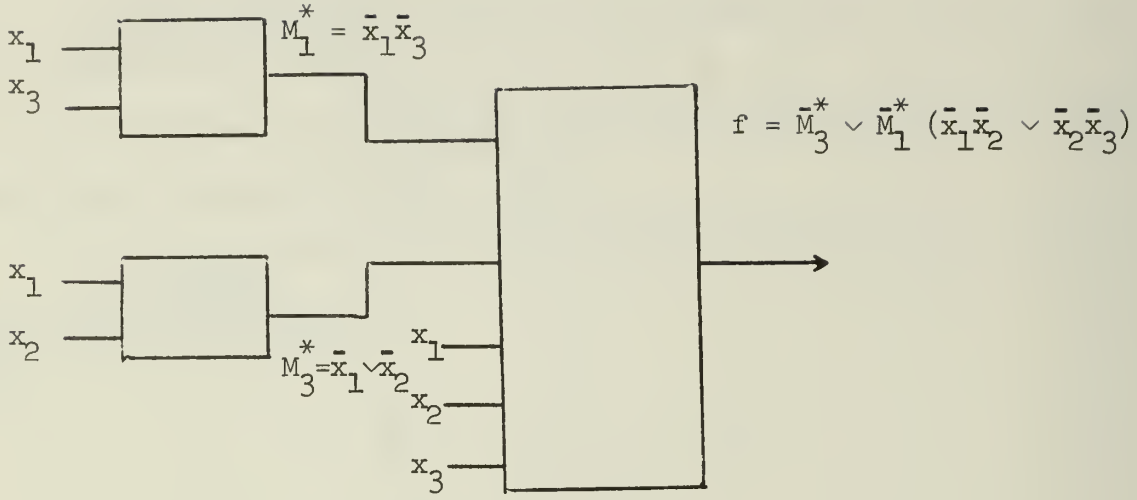


Fig. 6 A minimum network of f given by Table 6.

Up to this point, an entire procedure to obtain a minimum network with a single output has been presented. Next two sections will deal with a case in which multiple outputs are required and some topics related to the minimization of the number of connections and interconnections.

6. Minimum Multiple Output Networks

One of the advantages of our approach is that it can be easily extended to the synthesis of a mutiple output network. When we are to realize more than one non-negative function*, only necessary modification is that primitive additional columns e_i are generated in the negativity table for each given function. The rest of procedure, i.e., the generation of maximal compatible sets, the selection of a minimal compatible cover and the completion of each selected maximal compatible set, is exactly the same as that discussed so far. The procedure will also result in a network with the minimum number of gates.

Let us obtain a minimum network of two functions:

$$f_1 = x_1x_2 \vee x_1\bar{x}_3 \vee \bar{x}_1\bar{x}_2x_3$$

$$f_2 = x_1\bar{x}_2 \vee x_1\bar{x}_3 \vee \bar{x}_2x_3 .$$

	f_1	f_2	x_1	x_2	x_3	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	M_1^*	M_3^*
(1)	0	0	0	0	0	1	1	1	1	1					1	1
(2)	1	1	0	0	1	0									0	1*
(3)	0	0	0	1	0						1	1			1	1
(4)	0	0	0	1	1								1		0*	1
(5)	1	1	1	0	0		0								0	1*
(6)	0	1	1	0	1			0						1	0	1
(7)	1	1	1	1	0				0		0				0	0
(8)	1	0	1	1	1					0		0	0	0		
						1	1	**2**	1	1	1	1	1	1		
						2	2		2		2					

Table 10 Negativity table of f_1 and f_2

---

* When some of functions to be realized are negative, the synthesis would be simpler. In other words each function can be realized by a single gate. Furthermore, the outputs of gates for these negative functions can be used as inputs to other gates which are to realize non-negative functions. The modification of algorithm required is straightforward, however, and will not be given here.

The entries given underneath primitive additional columns e_1, e_2, ...,e_9 in Table 10 show from which of f_1 and f_2 the primitive additional column is generated. For example, the column e_1 is generated from both f_1 and f_2. The compatibility matrix is shown in Table 11. The generation of all the maximal compatible sets is also shown in Fig. 7.

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
e_2								
e_3								
e_4								
e_5								
e_6								
e_7								
e_8	×							
e_9	×	×	×					

Table 11 Compatibility matrix of f_1 and f_2

$$\{e_1\} \rightarrow \{e_1,e_2\} \rightarrow \ldots \rightarrow \{e_1,e_2,e_3,e_4,e_5,e_6,e_7\} \rightarrow$$

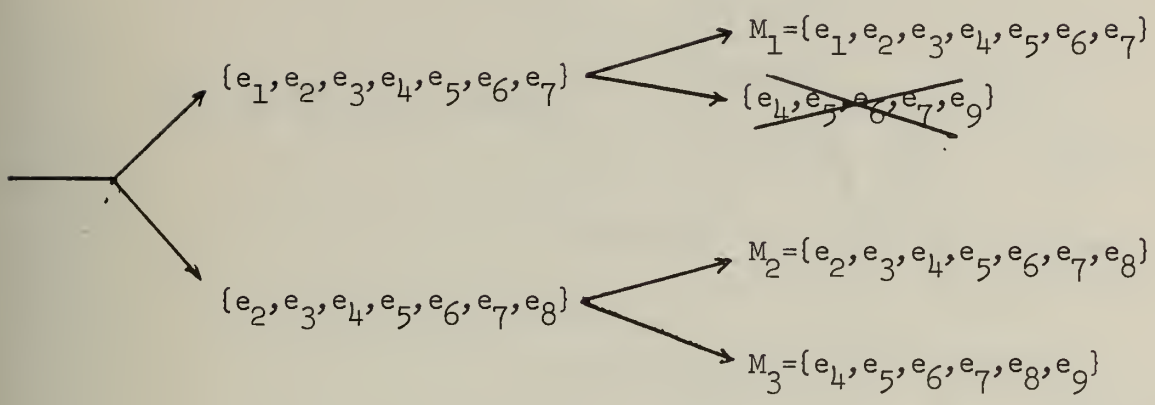


Fig. 7. Generation of maximal compatible sets of f_1 and f_2

Minimal compatible covers can be determined in the same way as the single output network problem. The covering table consists of 9 rows corresponding to $e_1, e_2, \ldots, e_9$ and 3 columns corresponding to M_1, M_2 and M_3. The minimal compatible cover for this problem is $\{M_1, M_3\}$. Negative completions of M_1 and M_3 are also shown in Table 10. M_1^* and M_3^* are written as

$$M_1^* = \bar{x}_1\bar{x}_3$$

$$M_3^* = \bar{x}_1 \vee \bar{x}_2 .$$

A minimum network of f_1 and f_2 obtained from those M_1^* and M_3^* is given in Fig. 8. f_2 does not require

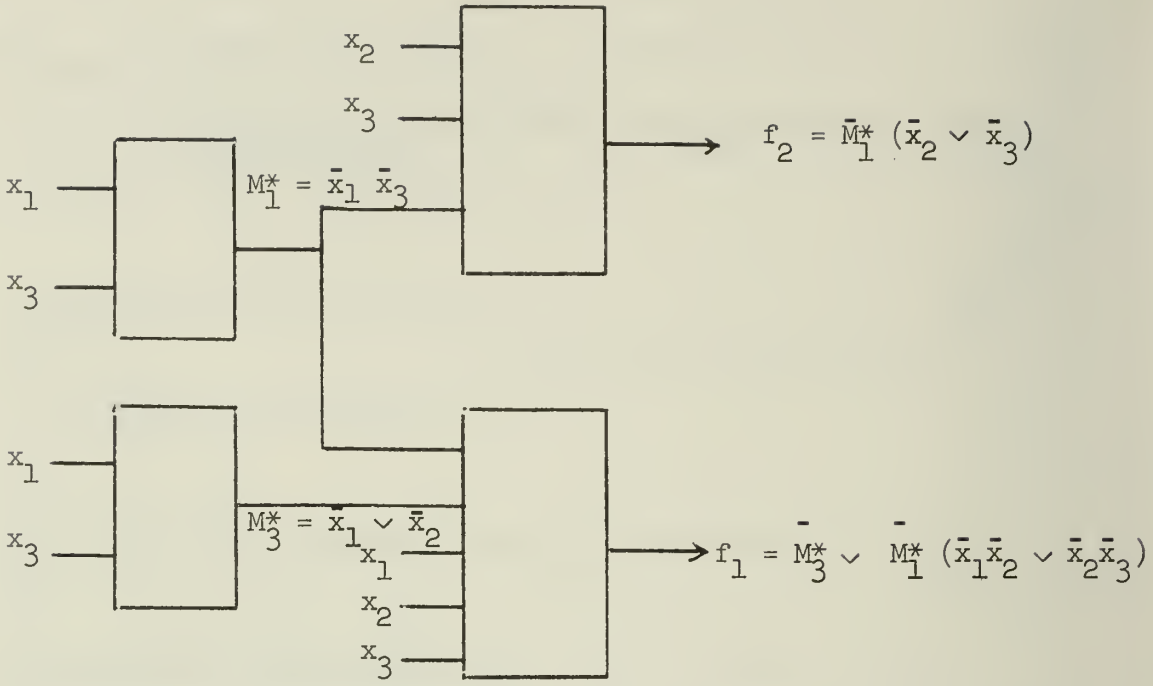


Fig. 8 Minimum network of f_1 and f_2

M_3^* because all the primitive additional columns e_1, e_2, e_3, e_4, e_6 generated from f_2 can be covered by a single maximal compatible set M_1. (A more rigorous discussion will be found in the next section.)

$$\{e_1\} \rightarrow \{e_1,e_2\} \rightarrow \ldots \rightarrow \{e_1,e_2,e_3,e_4,e_5,e_6,e_7\} \rightarrow$$

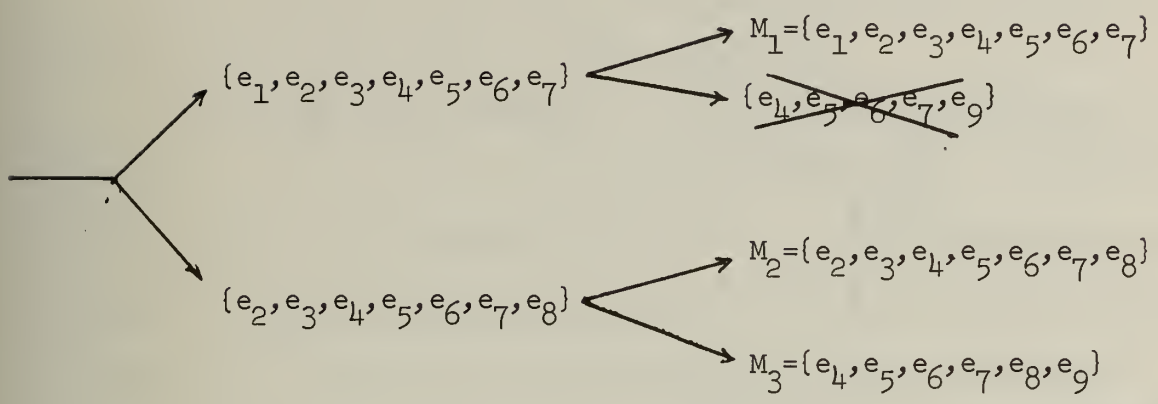


Fig. 7. Generation of maximal compatible sets of f_1 and f_2

Minimal compatible covers can be determined in the same way as the single output network problem. The covering table consists of 9 rows corresponding to $e_1, e_2, \ldots, e_9$ and 3 columns corresponding to M_1, M_2 and M_3. The minimal compatible cover for this problem is $\{M_1, M_3\}$. Negative completions of M_1 and M_3 are also shown in Table 10. M_1^* and M_3^* are written as

$$M_1^* = \bar{x}_1\bar{x}_3$$

$$M_3^* = \bar{x}_1 \vee \bar{x}_2 .$$

A minimum network of f_1 and f_2 obtained from those M_1^* and M_3^* is given in Fig. 8. f_2 does not require

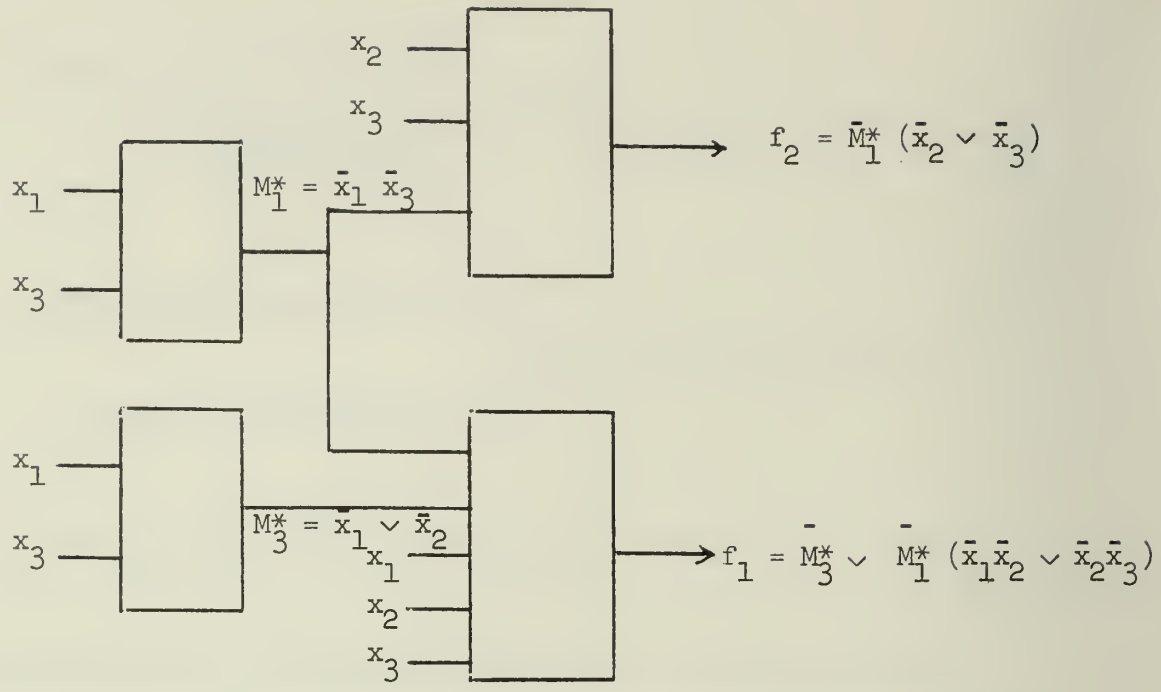


Fig. 8 Minimum network of f_1 and f_2

M_3^* because all the primitive additional columns e_1, e_2, e_3, e_4, e_6 generated from f_2 can be covered by a single maximal compatible set M_1. (A more rigorous discussion will be found in the next section.)

7. Some Considerations on the Reduction of Interconnections.

Although we can minimize the number of gates in a network by following the method presented so far, no consideration was given on the minimization of the number of interconnections.

The overall minimization of the number of interconnections (between the first and second levels) and connections of inputs to the first and second levels (from variables $x_1, x_2, \ldots, x_n$) seems difficult at present unless we exhaust all the possible networks and then pick up the best one among them. Here we will discuss two cases in which the number of interconnections and connections can be minimized in a restricted sense. One is minimizing the number of interconnections between the first and second levels, when designing a multiple output network. The other is the minimization of the number of connections from variables $x_1, x_2, \ldots, x_n$ fed to the first level.

Let us consider the former problem.

The minimality criterion is to find a network with the minimum number of interconnections between the first and second levels among those with the minimum number of gates.

Note that we are not concerned about the number of connections fed from the external variables $x_1, x_2, \ldots, x_n$.

The problem is trivial if the network has a single output, because it is then the same as minimizing the total number of gates in the network.

The following property is useful for our problem.

Theorem 7: Under the minimality criterion as stated above, there is a minimum network in which each gate in the first level represents a negative completion of maximal compatible set.

The proof is similar to that of Theorem 5. This theorem enable us to limit our consideration only to maximal compatible sets.

Before discussing the whole scheme of minimization, assume that a minimal compatible cover for functions f_1, f_2, ... f_q which are to be realized is already determined by the method described in Section 6. The minimal compatible cover consists of maximal compatible sets, $M_1, M_2, \ldots, M_v$. Then in the negativity table of f_1, f_2, ... f_q, it is shown which of primitive additional columns e_i are necessary for each function f_j (see the bottom row of Table 10). Every primitive additional column must be covered by some of maximal compatible sets chosen above. However, some function f_j may not require all the primitive additional columns for f_j to be negative and therefore some of maximal compatible sets may be omitted for f_j.

Assume that $e_{j1}, e_{j2}, \ldots, e_{jk_j}$, $j = 1, 2, \ldots, q$, are necessary in order for f_j to be negative. The interconnections from the first level to the output gate for each f_j can be minimized by selecting the minimum number of maximal compatible sets out of the minimal compatible cover $\{M_1, M_2, \ldots, M_v\}$ chosen for $f_1, f_2, \ldots, f_q$, such that each $e_{j\ell}$, $1 \leq \ell \leq k_j$, is covered by some of selected maximal compatible sets. Apply this process for $j = 1, 2, \ldots, q$. This is again a set-covering problem and can be solved similarly.

If the minimal compatible cover is unique, the network obtained above is the minimum network in the present sense. Otherwise, we have to apply the above procedure to every minimal compatible cover and then find the best one among those obtained. This procedure is not so inefficient as it appears.

Consider the example given by Table 10. This example has the unique minimal compatible cover $\{M_1, M_3\}$. First consider f_1. It requires primitive additional columns e_1, e_2, e_4, e_5, e_6, e_7, e_8, e_9 for f_1 to be negative as shown in Table 10. This means that f_1 needs both M_1 and M_3 to cover those 8 primitive additional columns. On the contrary, f_2 needs only e_1, e_2, e_3, e_4, e_6 in order to be negative, and these e_i are all included in one maximal compatible set $M_1 = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$. In fact, the realization of Fig. 8 shows that f_2 uses only $M_1^* = \bar{x}_1 \bar{x}_3$ although f_1 uses both $M_1^* = \bar{x}_1 \bar{x}_3$ and $M_3^* = \bar{x}_1 \vee \bar{x}_2$.

Next the same problem is formulated as an integer programming problem. Although the formulation seems neater than that given above, probably it will take more computation time, judging from the present status of integer programming.

Let us assume that $M_1, M_2, \ldots, M_p$ are all the maximal compatible sets for $f_1, f_2, \ldots, f_q$. Each f_j requires primitive additional columns $e_{j1}, e_{j2}, \ldots, e_{jk_j}$ in order to be negative. Let us introduce zero-one variables ϵ_{ji}, $j = 1, 2, \ldots, q$, $i = 1, 2, \ldots, p$. ϵ_{ji} assumes 1 if M_i is selected and actually used to cover some of $e_{j1}, e_{j2}, \ldots, e_{jk_j}$ for f_j, and 0 otherwise. Then prepare Table 12, by filling 1 in the entry $a_{k\ell,ki}$ if $e_{k\ell}$ is included in M_i, 0 otherwise, and $a_{k\ell,ji} = 0$ if $k \neq j$. Let us denote the matrix of these entries with A. Then consider the set of $\sum_{j=1}^{q} k_j$ inequalities with $p \cdot q$ variables ϵ_{ji}

$$A \vec{\epsilon} \geq \vec{1} . \tag{7.1}$$

	for f_1	for f_2		for f_q	
	$\epsilon_{11}\ \epsilon_{12} \cdots \epsilon_{1p}$	$\epsilon_{21}\ \epsilon_{22} \cdots \epsilon_{2p}$	. . .	$\epsilon_{q1}\ \epsilon_{q2} \cdots \epsilon_{qp}$	
e_{11} e_{12} ⋮ e_{1k_1}	$A^{(1)}$	0	. . .	0	≥ 1 ≥ 1 ⋮ ≥ 1
e_{21} e_{22} ⋮ e_{2k_2}	0	$A^{(2)}$	. . .	0	≥ 1 ≥ 1 ⋮ ≥ 1
.	.	.	.	.	
.	.	.	.	.	
.	.	.	.	.	
e_{q1} e_{q2} ⋮ e_{qk_q}	0	0	. . .	$A^{(q)}$	≥ 1 ≥ 1 ⋮ ≥ 1

Table 12. Illustration of integer program for minimization of interconnections.

This set of inequalities (7.1) requires that each $e_{k\ell}$ be covered by at least one of M_i which are to be selected. Note that $\sum_{i=1}^{p} \epsilon_{ji}$ represent the number of interconnections from the first level to the last gate for f_j.

Let us introduce new zero-one variables λ_i, $i = 1, 2, \ldots, p$, such that

$$\sum_{j=1}^{q} \epsilon_{ji} \leq q \cdot \lambda_i, \quad i = 1, 2, \ldots, p. \tag{7.2}$$

Then if one of ϵ_{ji}, $j = 1, 2, \ldots, q$, is 1, i.e., M_i is selected to cover some $e_{j\ell}$, then λ_i must assume 1. Otherwise (7.2) is not restrictive.

With these preparation, let us solve a integer linear programming (7.1) and (7.2) with the minimization of

$$U \sum_{i=1}^{p} \lambda_i + \sum_{i=1}^{p} \sum_{j=1}^{q} \epsilon_{ji},$$

where U is a sufficiently large positive number such that $U \geq p\ q$. It primarily minimizes $\Sigma\ \lambda_i$ which represents the number of maximal compatible sets actually used, and secondarily minimizes $\Sigma\ \Sigma\ \epsilon_{ji}$ which is the total number of interconnections between the first level and the second level. Obviously, this is exactly the same minimality criterion as we have discussed before. At present no computational experience has been obtained for this integer program.

Next let us discuss another minimization problem. For this problem, assume that we have already determined a minimal compatible cover for a given function (or functions). Each maximal compatible set will define a negative function of variables $x_1, x_2, \ldots, x_n$ which is then implemented as a gate in the first level.

Here we minimize the number of variables actually used for these gates, i.e., the total number of connections fed to the first level, assuming that maximal compatible sets for those gates in the first level are given. Contrary to the previous minimization problem, this assumption is necessary because compatible sets which are not maximal may reduce the number of connections further. However, the development of algorithm which eventually yields the network of the minimum number of interconnections and connections seems difficult at present. In this sense, the present procedure gives only an approximate solution to the problem.

Now let us consider a maximal compatible set in the given minimal compatible cover (the negative completion are not obtained yet) and the corresponding incompletely specified negative function g of variables $x_1, x_2, \cdots, x_n$.

Each pair of entries $g^{(j_1)} = 1$ and $g^{(j_2)} = 0$ are allowable because g is assumed to be negative. Denote all such pairs of entries as $P_1, P_2, \cdots, P_u$. Each pair P_i is allowable due to $x_{i1}, x_{i2}, \cdots,$ or x_{ik_i} (see Definition 4). Then the determination of the minimum number of variables for g is to select the minimum number of variables such that every P_i, $i = 1, 2, \cdots, u$, is allowable due to these selected variables. This is also a set-covering problem and therefore can be solved, though the detail is not given. This set-covering problem should be solved for every maximal compatible set corresponding to a gate in the first level.

As an example, consider the maximal compatible set M_3 derived from M_3^* of Table 9, by deleting the entries with stars. Table 11 shows the covering table of this problem. Each row represents a pair of entries P_i and each column represents a variable x_j. (For the above example, each pair is specified by a pair of row numbers of the column M_3.) The (i,j)th entry of the table is 1 if P_i is allowable due to x_j, and blank otherwise.

Table 13. Covering table for the determination of the minimum number of inputs to M_3 derived from M_3^* of Table 9.

P_i	x_1	x_2	x_3
(1.7)	1		
(1.8)	1		1
(3.7)	1		1
(3.8)	1		
(4.7)	1		
(4.8)	1		
(6.7)		1	
(6.8)		1	

Table 13 shows that we need only x_1 and x_2 to realize M_3 as a negative function of x_1, x_2, x_3. In fact, function $M_3^* = \bar{x}_1 \vee \bar{x}_2$ in Fig. 6 needs only x_1 and x_2.

Determination of inputs to the output gate can also be done similarly. In this case, however, we know the negative functions which are fed from the first level. Therefore, in determining external variables which are connected to the output gate, we need not consider pairs of entries which are already allowable due to those. The subsequent manipulation should be applied on this shrinked covering table.

8. Conclusion

The minimization of the number of gates in the network in which each gate can represent only negative function was converted into a set-covering problem. Since the set-covering problem is rather easy to solve, this approach appears helpful in obtaining such networks. Although some attempts are made, the minimization algorithm of the total number of interconnections and connections is still an open question.

It would be a direction of the future research to develop algorithms based on the other minimality criteria, such as the number of interconnections and connections, the total cost of gates which is evaluated differently from the number of gates, and also the extension of the approach in this paper to more general cases such as feed-forward networks.

REFERENCES

(1) S. B. Akers, "A truth table method for the synthesis of combinational logic," IRE Transaction on Electronic Computers, vol. EC-10, no. 4, pp. 604-615, December 1961.

(2) S. B. Akers, "Synthesis of combinational logic using three-input majority gates," Proceedings of the third annual symposium on switching circuit theory and logical design, pp. 150-157, October 1962.

(3) M. L. Balinski, "Integer programming: methods, uses, computation," Management Science, vol. 12, no. 3, pp. 253-313, November 1965.

(4) A. Cobham, R. Fridshal and J. H. North, "An application of linear programming to the minimization of Boolean functions," Proceedings of the second annual symposium on switching circuit theory and logical design, pp. 3-9, October, 1961.

(5) E. N. Gilbert, "Lattice theoretic properties of frontal switching functions," Journal of Mathematics and Physics, vol. 33, pp. 57-67, April 1954.

(6) T. Ibaraki, T. K. Liu, C. R. Baugh and S. Muroga, "An implicit enumeration program for zero-one integer programming," Report no. 305, Department of Computer Science, University of Illinois, January 1969.

(7) E. J. McCluskey, Introduction to the theory of switching circuits, McGraw-Hill Inc., New York, 1965.

(8) S. Muroga, Threshold logic, Lecture notes for EE 497 and EE 498, Department of Computer Science, University of Illinois, 1965-1966.

(9) E. Thomas and J. D. Callan, "Slow, but small, may win the race," Electronics, vol. 40, no. 4, pp. 179-182, February 20, 1967.

(10) Y. T. Yen, "A mathematical model characterizing four-phase MOS circuits for logical simulation," IEEE Transaction on Computers, vol. C-17, no. 9, pp. 822-826, September 1968.

FEB 7 - 1974

BOUND BY
HNM

UNIVERSITY OF ILLINOIS-URBANA
510.84 IL6R v.3 C002 v.308-315(1969)
TRANQUIL : a language for an array proce
3 0112 088404311